1. Prerequisites

Ensure you have the following installed:

- Python 3.9 or above
- Required Python libraries: pandas, numpy, matplotlib, mmh3

2. Download the Dataset

- Obtain the dataset (e.g., a list of commonly used passwords) and save it as a text file.
- Place the dataset file in the working directory or specify its file path in the code.

Passwods.txt

3. Code Structure

The project consists of the following components:

1. **HierarchicalHashFunction Class**: Implements the hashing mechanism using mmh3.
2. **CountingBloomFilter Class**: Implements the Counting Bloom Filter data structure.
3. **CuckooHashTable Class**: Implements the Cuckoo Hash Table data structure.
4. **Experimental Scripts**: Test and evaluate the data structures under different configurations.

Load the Dataset:

- In the script, specify the dataset file path:

file_path = 'passwords.txt'

Set Parameters: Configure the hyperparameters in the main script.
sizes = [10**5, 5 * 10**5, 10**6] # Sizes for the Bloom Filter and Hash Table

num_hashes_list = [3, 5, 7, 10] # Number of hash functions for testing

Execute the script in your Python environment:

1. The script will:
    - Load the dataset.
    - Initialize the Counting Bloom Filter and Cuckoo Hash Table with specified configurations.
    - Insert and query passwords, measuring performance metrics like insertion and lookup times.
2. View the Results:
    - The performance metrics will be printed in the terminal, including:
        - Insertion Time
        - Lookup Time
    - Visualizations will also be displayed, showing trends for different configurations.

5. Ensuring Reproducibility

To ensure consistent results across runs:

- A fixed seed (42) is used for the hash functions and random processes.

  random.seed(42)

  np.random.seed(42)